VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Analýza tvaru očních víček v obrazech

Shape Analysis of Eyelid in Images

2017

Petr Kábrt

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **Petr Kábrt**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R059 Mobilní technologie

Téma: Analýza tvaru očních víček v obrazech
Shape Analysis of Eyelid in Images

Jazyk vypracování: čeština

Zásady pro vypracování:

Sledování únavy a chování řidiče v automobilech se v posledních letech stává velmi diskutovaným tématem. S tím souvisí i analýza tvaru očí v obrazech, kdy se právě tato analýza (zejména detekce tvaru očních víček) může stát nosným prvkem vyhodnocování otevřených nebo zavřených očí.

1. Seznamte se se základními pojmy v oblasti detekce objektů a jejich tvarů v obrazech.
2. Seznamte se s knihovnou OpenCV.
3. S pomocí knihovny OpenCV jedno vhodné řešení implementujte.
4. Experimentálně ověřte funkčnost, přesnost a rychlost navrženého detektoru.
5. Své závěry řádně zdokumentujte v textu práce.

Seznam doporučené odborné literatury:

Liya Ding and Aleix M. Martinez: Precise detailed detection of faces and facial features, in Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 24-26, 2008.

Tae-Hong Min and R. H. Park: Comparison of eyelid and eyelash detection algorithms for performance improvement of iris recognition, 15th IEEE International Conference on Image Processing (ICIP), San Diego, pp. 257-260, 2008.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Radovan Fusek, Ph.D.**

Datum zadání: 01.09.2016

Datum odevzdání: 28.04.2017

doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta elektrotechniky a informatiky 1D16 OSTRAVA

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 28. dubna 2017 .....

Rád bych poděkoval vedoucímu bakalářské práce Ing. Radovanu Fuskovi, Ph.D. za odbornou pomoc a konzultaci při vytváření této práce.

Abstrakt

Tato bakalářská práce se zabývá zkoumáním tvaru očních víček v obrazech a následnému využití těchto získaných dat. V úvodu je rozepsán důvod této detekce, tím je především únava řidiče na cestách a problémy, které mikrospánek může způsobit. Dále práce rozebírá různé techniky, kterými lze této detekce dosáhnout a popisuje jejich využití. V praktické části je znázorněna implementace vlastní metody, která byla vytvořena pro řešení daného problému. Ta byla inspirována metodami použitými v již dříve popsaných technikách.

Klíčová slova: analýza, oční víčka, únava, řídič, bakalářská práce

Abstract

This bachelor thesis deals with detecting eyelid shape in images and followed-up usage of this collected data. The introductory part shows a reason for this detection which is mostly driver's drowsiness on the road and problems which microsleep can cause. Next, the thesis analyses different techniques that can be used to achieve this detection and describes their usage. The practical part of this thesis shows the implementation of the custom method, created to solve this exact problem. This method was inspired by the methods described in the introductory part.

Key Words: analysis, eyelids,drwosiness,driver, bachelor thesis

Obsah

Seznam použitých zkratek a symbolů

FPS	–	(Frames Per Second) Snímky za sekundu
OpenCV	–	(Open Source Computer Vision Library) Otevřená volně šiřitelná knihovna pro zpracování obrazu počítačem pro jeho následnou analýzu
SQDIFF	–	Square difference
TM	–	(Template Matching) Porovnávání vzorů

Seznam obrázků

Seznam tabulek

1 Úvod

V dnešním světě, ve kterém každá rodina vlastní jeden nebo dva automobily, se stává únava řidičů během jízdy stále větším problémem. I když ospalost může vypadat na první pohled nevinně, její efekty se dají srovnávat s efekty alkoholu. Při kontrole po zastavení řidiče však nelze mikrospánek na rozdíl od požití alkoholu rozpoznat. Není tedy překvapující, že ospalost může až za třetinu dopravních nehod.

Proto se stává vývoj opatření proti únavě stále větší prioritou. Základem všech těchto opatření je detekce stavu řidiče během jízdy, aby byl včas varován a mohl dostatečně rychle reagovat. Jedním z nejvýraznějších příznaků únavy je pomalé přivírání očí řidiče, zpomalené mrkání nebo prodloužení intervalu zavření očí.

Tato práce se tedy zaměřuje na seznámení s různými detekčními postupy pro detekci očí a jejich následné využití pro vytvoření požadovaného detektoru, který by se zaměřoval na detekci očních víček a jejich tvaru. Následně se tato práce snaží detektor upravit tak, aby byl schopný získat z obrazu dostatek informací, podle kterých může rozpoznat, v jakém stavu se oči nacházejí.

2 Způsoby detekce tvaru očních víček

Detekce objektů je proces, při kterém se v obraze či videu hledají instance objektů z reálného světa. Těmito objekty mohou být například lidské tváře, budovy nebo dopravní značky. Detekce objektů typicky využívá extrahování vlastností daného objektu a učících se algoritmů pro nalezení instance objektové kategorie. Detekce objektů se běžně využívá u bezpečnostních aplikací, monitorovacích systémů nebo v systémech pro automatické parkování.

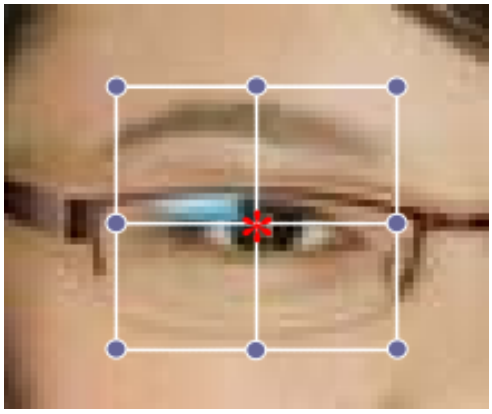
Tato práce se konkrétně zaměřuje na detekci očních víček v obrazech. Detekce očí hraje velkou roli v řadě programů různého využití. Zatím je nalezení techniky, která by měla dokonalou účinnost prakticky nemožné. Detekci totiž ovlivňuje velké množství faktorů. I když se většina očí zdá na první pohled stejná, při bližším průzkumu, při kterém byly oči z obrázků ořízнuty, je zřejmé, že tomu tak není. Oči mohou mít velmi unikátní kombinaci vlastností, těmi jsou velikost zornice a barva (kůže očního víčka i duhovky). Dále je detekce velmi ovlivněna tím, zda daná osoba nosí brýle a v jakém úhlu a intenzitě na ni dopadá světlo. V následujících podkapitolách budou nastíněny možné postupy pro detekci očních víček.

2.1 Detekce založená na hledání vzoru (template matching)

V tomto článku autoři Liya Ding a Aleix M. Martinez [1] popisují techniku, pomocí které získali přesnosti dosahující až 6.2 pixelů při detekci tváře a očních víček. Dosáhli toho tak, že nejprve detekovali tvář dané osoby pomocí nalezení vlastností tváře, jako jsou oči, nos a ústa. Vzorky, které alespoň vzdáleně odpovídaly předlohám, pak byly detekovány jako tvář. Tato technika dovoluje detekci tváře i za extrémních podmínek. Algoritmus se pro zvýšení přesnosti detekce musel naučit rozpoznávat detekovanou tvář od jejího okolí. Tím byla zajištěna i následná přesnost. Dále algoritmus mohl prohledat nalezenou oblast pro výskyt očí, zde nalezl jejich střed, koutky, zornici a nakonec určil i tvar očních víček. V následujících podkategoriích je popsána každá z těchto technik.

2.1.1 Detekce středu oka podle hledání vzoru

Pro pozici očí byl vytvořen opět vlastní klasifikátor, který porovná oko s databází tisíců vzorků. Pokud oko neodpovídá žádnému ze vzorků, je vrácen přibližný odhad. Toto je většinou způsobeno různými přechody světla nebo doplňky, např. brýlemi, které mohou zmást klasifikátor. Aby se tomuto problému zabránilo, klasifikátor byl trénován, aby nacházel sousední oblasti oka a tím táhl detekční oblast více k předpokládané lokaci středu. Algoritmus tedy porovnává oblast oka s 941 vzorovými obrázky ořízнutých očí a poté nalezenou oblast porovná se 7 528 obrázky okolních oblastí oka. Výsledek pak vytvoří jakousi mapu, podle které lze určit střed. Obrázek 1 ukazuje, jak tato technika funguje.



Obrázek 1: Modré body jsou středy pasujících obrázků okolních oblastí oka a červená hvězda značí nalezený střed. Vzdálenost mezi středem a krajními body je 24 pixelů.

2.1.2 Detekce koutků pomocí hledání vzoru

Nyní se může začít hledat tvar oka, respektive očních víček. Po nalezení středu je potřeba nalézt alespoň dva další záchytné body. Těmi se stanou koutky daného oka. Ty se hledají tak, že se zopakuje stejný postup jako pro nalezení středu. Zde je však zaměřen na levý a pravý koutek. Stejná technika musí být použita z toho důvodu, že koutky očí samy o sobě mají taktéž velký počet tvarů, textur a barev. Detektor také musí rozpoznat koutek oka při použití make-upu nebo když daná osoba nosí brýle. Projde se tedy opět databáze vzorových obrázků okolních oblastí koutků a sestaví se jejich střed. Algoritmus se naučí zpracovávat koutky pouze jednoho oka. Pro detekci koutků na druhém oku se obraz jednoduše převrátí díky symetrii obličeje.

2.1.3 Detekce zornice pomocí gradientu

Nastává jeden z posledních kroků, detekce zornice. Pro tento krok se obraz převede do odstínů šedi. V tomto režimu je zornice skoro pokaždé nejtmavším bodem oblasti oka (nehledě na barvu duhovky). Aby se mohla vyloučit chybná detekce kvůli faktorům, jako je tmavý oční stín nebo make-up, jsou hledána lokální minima, která mají největší gradient mezi každým lokálním minimem a okolím. Největší gradient je tedy získán pouze tehdy, pokud se detektor pohybuje z tmavé oblasti duhovky do světlé oblasti bělma. Díky tomu lze vytvořit velmi robustní detektor.

2.1.4 Detekce tvaru víček pomocí gradientu

Zatímco duhovka a zornice jsou tmavé oblasti, oční víčka mají barvu kůže. Toto je obzvláště pravda u spodního víčka, které má jen velmi omezenou pohyblivost a není tedy tolik ohrožené deformací, stíny a dalšími rušivými vlivy. Stále však musí být předpokládány problémy jako make-up či brýle. Proto se zde opět za použití detektoru největšího gradientu vedou z jednoho koutku oka do druhého křivky podél očních víček. Křivka, která má největší počet těchto bodů,

je pak vybrána jako nejpřesnější tvar daného očního víčka. Výsledek této detekce je znázorněn na obrázku 2

Obrázek 2: Hvězdičky znázorňují koutky očí. Duhovka je znázorněna jako kruh, ve kterém červená část znázorňuje viditelnou oblast. Horní a dolní kontury očních víček jsou znázorněny modře.

2.1.5 Klady a zápory

Nepopiratelnými klady pro tuto techniku jsou určitě její přesnost a robustnost. Tato technika dokáže najít vyhledávané rysy a detailně je určit i za nepříznivých podmínek. Testované obrázky byly zároveň předloženy několika lidským subjektům, které měly za úkol vyznačit vyhledávané rysy podle daných instrukcí. Výsledky softwaru měly průměrnou chybovost 2.1%, zatímco manuální detekce měla průměrnou chybovost 1.5%. Tento rozdíl je však v celkovém pohledu zanedbatelný a ukazuje na velmi dobrý výkon algoritmu. Nevýhodou tohoto algoritmu však mohou být kvůli nutnému porovnání s velmi početnou databází vzorů nároky na výpočetní výkon.

2.2 Metoda založena na hledání kontur

V tomto článku se autoři Vladimir Vezhnevets a Anna Degtiareva [2] rozhodli pro jiný postup. Tento článek již počítá s předešlou detekcí oblasti tváře a pracuje s již pouze oříznutou oblastí oka. Detekce zbývajícího modelu se pak provádí na barevném obrázku ve čtyřech krocích:

- Detekce středu duhovky
- Odhad poloměru duhovky
- Detekce kontur horního víčka
- Odhad dolního víčka

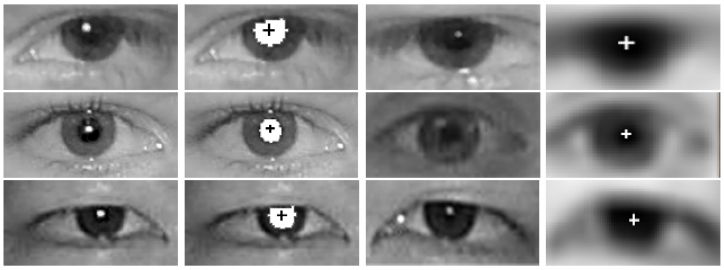
2.2.1 Detekce středu duhovky pomocí shluků

Nejdříve se nalezne odhadovaný střed duhovky tak, že se střední část oka prohledá pro shluky pixelů s maximální hodnotou (pixely obsahující největší hodnotu červeného kanálu) a porovná je s předem určenou hranicí. Z nalezené oblasti se vypočte střed, okolo kterého se provede kruhové vyhledávání podle přibližného poloměru zornice. Zde se testují 3 podmínky:

1. Nalezené pixely by neměly mít menší minimum než nastavený globální práh pro oko. (Přidání tmavých pixelů do výběru.)

2. Rozdíl mezi nalezenými pixely by neměl být menší než předem nastavený práh. (Přidání světlých pixelů do výběru způsobeného odrazem světla od zornice)

3. Číslo tmavých pixelů by nemělo být menší než předpokládané minimum.



Obrázek 3: Přibližný odhad středu zornice podle červeného kanálu (a, c). (b) – oblast výběru zornice, znaménko plus znázorňuje přibližný střed. (d) – obraz po aplikování minimum filtru s označením středu shluku nejtmavších pixelů.

Všechny oblasti, ve kterých jsou tyto podmínky splněny, jsou zprůměrovány pro nalezení očekávaného středu oka. Obrázek 3 (b) zobrazuje typický výsledek. Pokud není nalezena žádná vyhovující oblast, aplikuje se 5x5 minimum filtr. Tento filtr nejprve zkontroluje oblast o velikosti 5x5 pixelů a následně nahradí centrální pixel této oblasti nejmenší hodnotou. Poté je na oko aplikována metoda, pomocí které se hledají shluky tmavých pixelů v kruhovém tvaru v centru konvoluce:

$$W(x, y, c) = \frac{\sin((x^2 + y^2)/c}{(x^2 + y^2)/c} \tag{1}$$

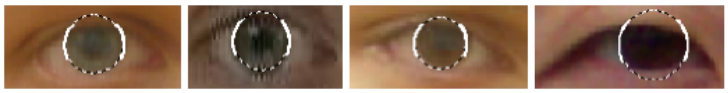
Tato funkce se snaží přiřadit hodnotu kruhové oblasti ve středu a na zbytek přidat záporné hodnoty. Po konvoluci výsledných 5% tmavých pixelů určuje přibližný střed, viz obrázek 3 (d).

Nyní, když byl nalezen přibližný odhad, lze zjistit přesný tvar duhovky pomocí nalezení přesného středu (x_c, y_c) a poloměru (r). Toho lze dosáhnout pomocí použití následujícího algoritmu:

$$f_\Theta(x_c, y_c, r) = \int_{\theta \epsilon \Theta} I(x_c + rcos\theta, y_c + rsin\theta)d\theta \tag{2}$$

Tento algoritmus pracuje na základě dvou předpokladů. Duhovka je přibližně kruhového tvaru a hledaný střed a poloměr leží v kruhu, který je tmavší než jeho pozadí. $I(x, y)$ je obrázek oka a Θ je v rozsahu $[0, 2\pi]$. Vzhledem k tomu, že duhovka je většinou zastíněná dolním

nebo horním víčkem, musíme omezit $\phi = [-\pi/4, \pi/6]U[5\pi/6, 5\pi/4]$ a prohledat okolí předem odhadnutého středu pro poloměr.



Obrázek 4: Výsledek detekce duhovky. Kruh označuje vypočtenou oblast, bílá část kruhu označuje část, která sloužila k výpočtu poloměru.

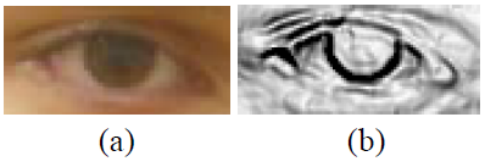
2.2.2 Detekce očního víčka

Hledaný model oka se skládá z křivky horního víčka a z křivky dolního víčka. Střed duhovky a její poloměr byl určen již v předešlém kroku. Horní víčko se detekuje ve třech krocích.

1. Nalezne se set bodů patřících hornímu víčku.
2. Tento set je prozkoumán, zda se nejedná o body ležící mimo víčko.
3. Body se propojí křivkou pro vytvoření potřebného tvaru.

Pro dolní víčko se křivka napojí z jednoho koutku oka do druhého a její zakřivení se určí podle bodu nacházejícím se na nejnižším bodu z dříve vypočteného kruhu duhovky.

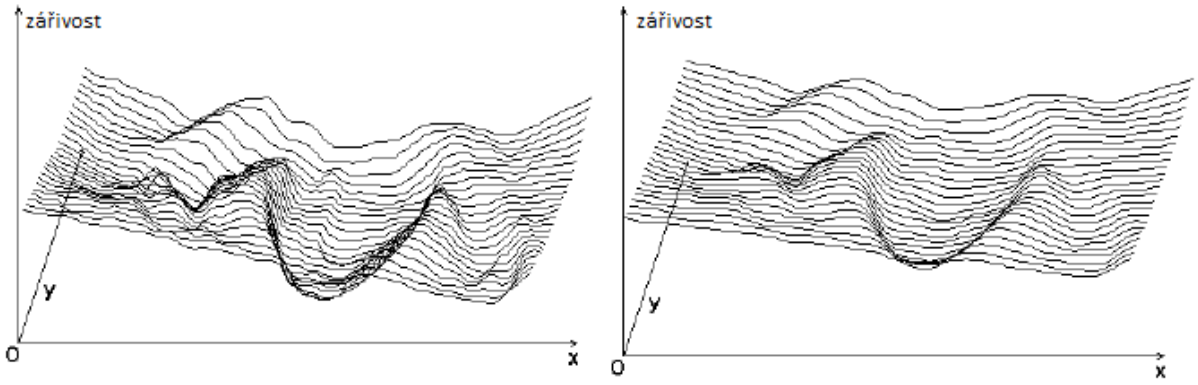
Potřebné body lze nalézt tedy pomocí mnoha již přístupných technik. Jednou z nich je detekce hran. Ta by fungovala dostatečně dobře v jiných případech, zde ovšem je několik problémů. Jedním z nich je velké množství nevyžádaných hran, které způsobují nepřesnosti, dalším je problém s nasvícením oka. Velmi často se totiž stává, že mezi spodním víčkem a bělmem není dostatečný gradient pro vytvoření hrany. Detektor tedy považuje danou plochu za jednotný celek. Díky těmto dvěma hlavním problémům se stává tato technika nepoužitelnou. Je však možné využít některé její vlastnosti a vytvořit detektor hran vhodný pro využití v tomto algoritmu. Níže uvedený obrázek 5 ukazuje, jak detektor hran funguje.



Obrázek 5: Ukázka obrázku oka (a) a jeho mapy hran založených na gradientu (b). Tmavší pixely korespondují s oblastmi s větším gradientem.

Autoři článku se tedy rozhodli pro algoritmus, který bude zkoumat obrázek po řádcích a bude v nich hledat lokální minima změn v nasvícení. Při bližším prozkoumání se totiž zjistilo, že jejich lokace odpovídá koutkům oka. Lze tedy oskenovat celý obrázek touto metodou a získat 3D mapu, viz obrázek 6, se kterou následně můžeme pracovat. Získaná data jsou většinou plná drobných chyb, proto se zavádí předzpracovávající metoda obrazu, která pomůže vyhladit graf

a zbavit ho těchto chyb. Daná metoda se skládá z několika různých filtrů, které se aplikují na obrázek ještě před tvorbou grafu. Jsou jimi mediánní filtr a filtr s dolní propustností.

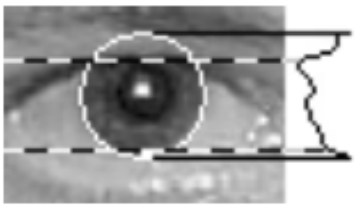


Obrázek 6: Pseudo 3D graf nasvícení stejného obrázku oka, který byl použit pro vytvoření mapy hran v obrázku 5. Levý graf - hrubá data z obrázku. Pravý graf – předzpracovaná obrazová data.

Nyní, když máme graf, můžeme přistoupit na další kroky této nově zvolené metody: určení výšky otevření oka, detekci bodů koutku oka, eliminaci přebytečných bodů a napojení křivky.

2.2.3 Detekce výšky otevření oka

Výška otevření oka se detekuje pomocí skenování oblasti duhovky shora dolů. Na každém řádku se spočte průměrná zářivost červené barvy. Oblast, kde se nachází minimální červená barva, je oblast duhovky, která není zastíněná očním víčkem.

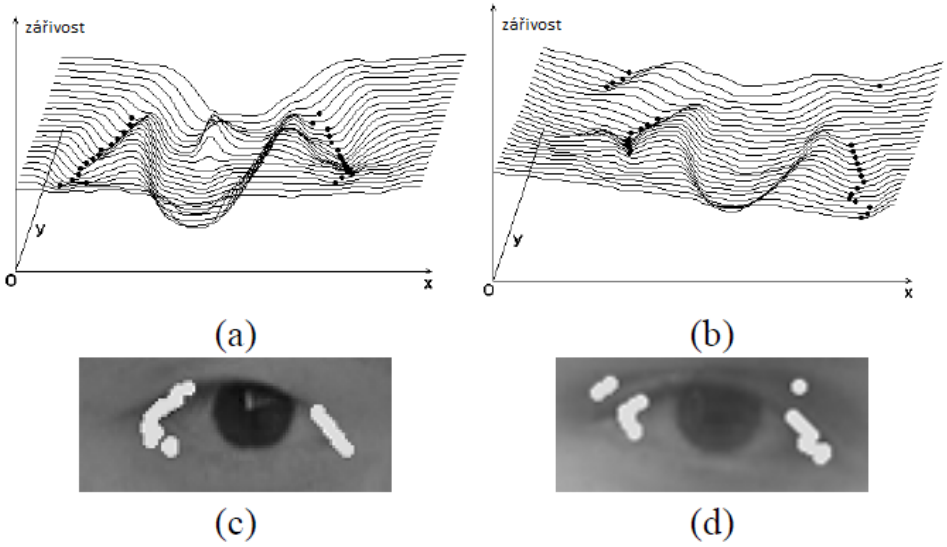


Obrázek 7: Detekce výšky otevření oka pomocí řádkové analýzy průměru zářivosti červené barvy.

Tato oblast nyní slouží jako oblast, ve které se hledají body koutků. Pokud detekce funguje správně, body by se měly zobrazit pouze na horním víčku. V případech, kdy se například špatně vypočte poloměr duhovky, budou body zobrazeny i na spodním víčku. Proto hledání koutků začíná až tehdy, když se průměrné množství zářivosti červené barvy v řádcích začne snižovat.

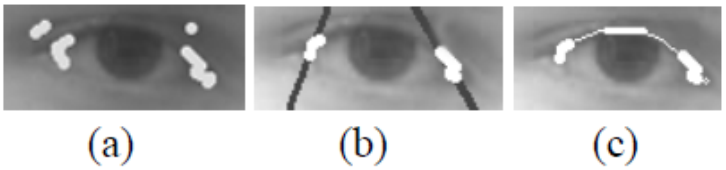
2.2.4 Eliminace přebytečných a chybných bodů

Předešlá detekce může získat i chybné body, které by snížily přesnost výsledné křivky. Tyto body se většinou nacházejí mimo horní víčko nebo dokonce mimo oblast oka, viz obrázek 8(d). K odstranění těchto přebytečných bodů se na levou a pravou skupinu bodů přichytí dvě přímky



Obrázek 8: (a,b) – Pseudo 3D graf nasvícení dvou obrázků oka s vyznačením koutků černými body. (c, d) – obrázky očí ve stupních šedi s vyznačenými koutky bílými body.

pomocí Houghovy transformace. Bylo prokázáno, že ta funguje velmi dobře i za špatných podmínek a ignoruje přebytečný šum. Funguje totiž na základě parametrické reprezentace objektu, který má být detekován. Jinými slovy, pokud chceme detekovat úhel očních víček, můžeme upravit Houghovu transformaci tak, aby hledala pouze body tvořící přímku v daném směru a úhlu. Algoritmus vytvoří několik přímek. Vybrána je ta, která odpovídá určitým předpokladům a obsahuje alespoň 30% bodů z vybrané podmnožiny. Všechny ostatní body jsou poté odstraněny, viz obrázek 9(b), a z výsledných bodů se nyní dá určit, v jakém úhlu se má napojit výsledná křivka.



Obrázek 9: (a) - Veškeré získané body, včetně chybných. (b) – Podmnožina bodů získána využitím Houghovy transformace včetně přímky kterou tvoří. (c) – Čtvercová mnohočlenná křivka přichycená na finální množinu bodů.

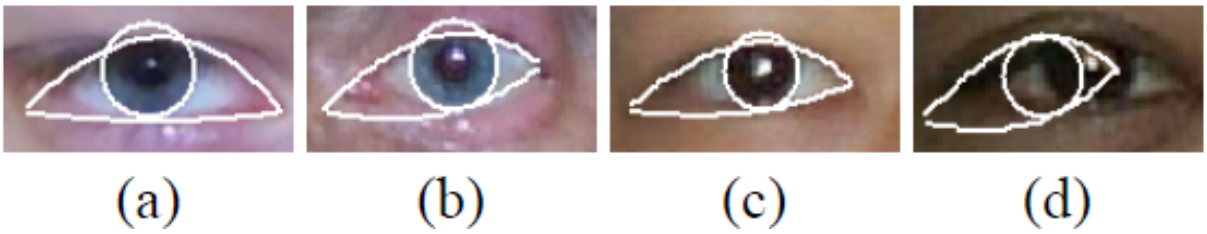
2.2.5 Přichycení křivky na tvar víčka

Ze zbývajících bodů se vybere ten, který se nachází nejvíce vlevo a nejvíce vpravo. Tyto body fungují jako přesná lokace koutků oka. Všechny ostatní body od těchto dvou nahoru jsou součástí horního očního víčka. Nakonec se ještě přidají body znázorňující místo, na kterém se horní víčko dotýká duhovky. Výslednou skupinou bodů se povede křivka, která se na body horního víčka

přichytí a kopíruje jejich tvar. Dolní víčko se detekuje pomocí křivky, která vede z koutků oka a následně prochází nejnižším bodem duhovky. Výsledek této metody je zobrazen na obrázku 10 níže.

Obrázek 10: Běžně získané výsledky

Tato detekce ovšem není bezchybná. Často se stává, že při špatné kvalitě zdrojového obrázku se kontura oka příliš roztáhne a algoritmus určí pozice bodů špatně. To obvykle způsobí posunutí celé detekce doleva jako na obrázku 11 níže, přesněji snímky a až c. Snímek d ukazuje nejzávažnější problém, a to selhání při detekci duhovky. V tomto případě detekce selže kompletně v dalších krocích.



Obrázek 11: Výsledky se špatnou nebo nepřesnou detekcí

2.3 Další způsoby detekce

Jak již bylo zmíněno dříve, způsobů detekce existuje opravdu mnoho. Kvůli délce této práce je ovšem nebylo možné uvést všechny. V metodě vyvinuté Amit Madhukar Waghem a Satish R Todmalem [3] bylo využito Houghovy transformace k nalezení víček. Další metody se zabývaly například využitím tvarového vzoru [4], detekcí hran [5], přichycením parabolické křvky [6] nebo zkombinováním hranového detektoru a Houghovy prostorové transformace [7].

3 Tvorba výsledného detektoru

Výsledný detektor byl vytvořen po dlouhém experimentování s různými technikami nastudovanými z vědeckých článků, z nichž ty nejdůležitější jsou zmíněny v předešlých kapitolách. Nyní následuje postup, který vedl k výsledné tvorbě funkčního detektoru s relativně velkou přesností. Následující kapitoly popisují knihovnu, která byla použita, videa, která byla testována a společné základní kroky, které byly provedeny u každé verze detektoru. Následně jsou popsány předešlé verze výsledného detektoru a důvody, proč byly nevyhovující.

V kapitole 3.5 Detekce pomocí hran je verze detektoru, která využívala Cannyho detekci hran společně s Houghovou transformací.

Kapitola 3.6 Detekce pomocí porovnání vzoru popisuje využití TM (Template Matching) techniky, neboli porovnání vzorů k určení základních bodů a určení tvaru víček pomocí změny barev.

Třetí kapitola 3.7 Detekce pomocí kontrastu popisuje třetí a konečnou verzi detektoru, které bylo dosaženo pomocí převzetí určitých částí z předešlých dvou pokusů a jejich úpravou k tvorbě funkčního detektoru.

Každý z výsledných detektorů obsahuje výslednou tabulku, např. tabulka 1, ve které byla procenta zaokrouhlována na celá čísla. Sloupce "Špatný tvar" a "Špatný stav" ukazují, kolik snímků bylo špatně odhadnuto softwarem oproti vizuální kontrole.

3.1 Knihovna OpenCV

OpenCV [8] je otevřená volně šiřitelná knihovna pro zpracování obrazu počítačem a pro jeho následnou analýzu. Tato funkce je použita například při nalézání objektů v obraze, rozpoznání tváří, různých činností či následování objektu a zaznamenávání jeho trasy. To vše se většinou provádí v reálném čase, mohou však být analyzována i videa či jednotlivé obrázky. V knihovně se momentálně nalézá přes 2500 optimalizovaných algoritmů, které obsahují klasické i nejnovější algoritmy pro počítačovou vizi a strojové učení. Díky těmto vlastnostem byla tato knihovna vybrána jako hlavní nástroj pro tvorbu následujících detektorů.

3.2 Testovaná videa

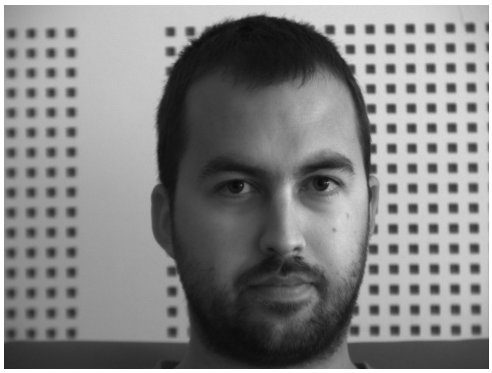
K otestování následujících technik a algoritmů byla využívána především následující videa, která se lišila v kvalitě, světelných podmínkách a délce. Výsledná videa, která byla produkována pro porovnání, jsou ve stejné kvalitě jako videa předchozí. Jediný rozdíl je, že se jejich rychlost sníží na 10 FPS pro lepší vizuální kontrolu. Toto číslo je však možné jednoduše změnit na standardních 24 FPS v kódu. Zde je náhled testovaných videí.

Obrázek 12: Video 1

3.2.1 Video 1

První video (obrázek 12) bylo použito kvůli nasimulování běžně dostupné web kamery. Rozlišení videa s velikostí 1024x768 je menší než tradiční HD. Rychlost videa je 29 FPS. Video obsahuje 264 snímků, je tedy dlouhé přibližně 8 sekund. Ve videu je zachyceno mrkání a pohyb zornice všemi směry. Video není nijak barevně upravované.

3.2.2 Video 2



Obrázek 13: Video 2

Druhé video (obrázek 13) bylo použito pro otestování detektoru na HD kameře. Rozlišení videa je tedy 1280x960. Toto video je oproti ostatním videím zpomalené na 13 FPS, což s jeho délkou 395 snímků tvoří přibližně 28 sekund. Obsah videa se nijak vizuálně neliší od videa 1, ale jeho zvýšená kvalita nutí algoritmus zkoumat mnohem větší plochu. Video je převedené do odstínů šedi.

3.2.3 Video 3

Třetí video (obrázek 14) je video s nejhorší kvalitou. Představuje běžně dostupné kamery v chytrých telefonech s cenovým rozpětím do 5000 Kč. Video má rozlišení 640x368 a bylo natočeno

Obrázek 14: Video 3

v 30 FPS. Obsahuje 512 snímků, což s touto rychlostí tvoří přibližně 17 sekund. Z pozadí postavy je video silně nasvícené a obsahuje podobné pohyby očí jako předešlá dvě videa. Video nebylo nijak barevně upravené.

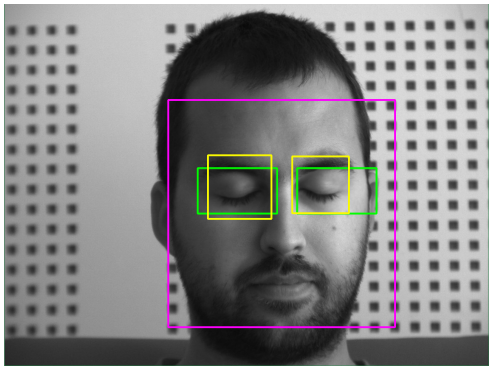
3.3 Hledání pozice oka

Vzhledem k tomu, že každý z následujících detektorů pracuje s již nalezenou oblastí oka, určuje se tato pozice stejně pro všechny.

Nejdříve se v obrazu nalezne tvář pomocí Haarova klasifikátoru určeného pro tento typ detekce. Ten je nastaven tak, aby hledal dostatečně velké tváře. Tímto se výrazně zrychlí následná detekce.

První metoda využívá detekovanou oblast obličeje a vytvoří dvě oblasti podle předem zadaných hodnot. Většinou se předpokládá, že vrchní část levého oka se nachází 30% vzdálenosti od vrchní hranice tváře a přibližně 13% od levého okraje. Šířka je nastavena na 35% a výška na 20%. Tyto oblasti jsou využité především k tomu, aby se zabránilo případné chybné lokalizaci oka.

Druhá metoda využívá již předepsaný Haarův klasifikátor na detekci očí pro prohledání oblasti nalezené tváře. Pokud žádná tvář není nalezena, například z důvodu zakrytí důležité části obličeje, klasifikátor prohledá celý obraz. To ovšem zpomalí výslednou detekci kvůli nutnému procházení snímku po mnohem menších oblastech. Výsledek nalezení této pozice je vidět na obrázku 15.



Obrázek 15: Výsledek detekce pozice oka. Růžový obdélník - tvář detekovaná pomocí Haarova klasifikátoru. Zelený obdélník - předpokládaná oblast oka. Žlutý obdélník - oko detekované pomocí Haarova klasifikátoru.

3.4 Základní společné kroky

Předtím než se využije jakýkoliv z níže uvedených kroků, je nutné provést určité základní kroky, které sdílí všechny následující pokusy. Všechny tyto kroky budou znázorněny na následujícím snímku - obrázek 16.

Obrázek 16: Testovací snímek tváře před úpravami.

3.4.1 Převod do odstínů šedi

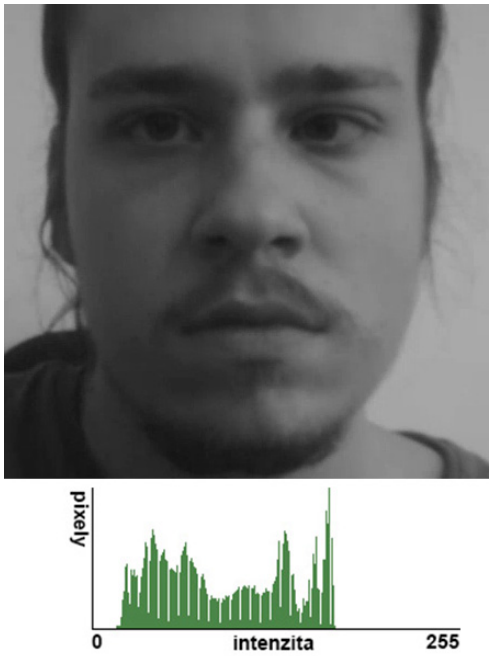
Nejprve se obraz převede do odstínů šedi. To se udělá tak, že se hodnoty všech kanálů vynásobí danou konstantou a výsledná hodnota se uloží do výsledné pozice pixelu v matici (Y).

$$Y \leftarrow 0.299 * R + 0.587 * G + 0.114 * B \tag{3}$$

Nově vytvořená matice obsahuje kopii obrazu v odstínech šedi, se kterou detektor pracuje. Obraz musí obsahovat přesnou výšku a šířku originálu, aby se výsledné vizuální prvky jako detekční body zobrazily na správném místě.

3.4.2 Vyrovnání histogramu

Dalším krokem pro zvýšení přesnosti je vyrovnání histogramu. To zaručí, že se vyrovná světlost snímku a zvýší se jeho kontrast. Algoritmus nejprve vypočítá histogram vloženého snímku - obrázek 17.



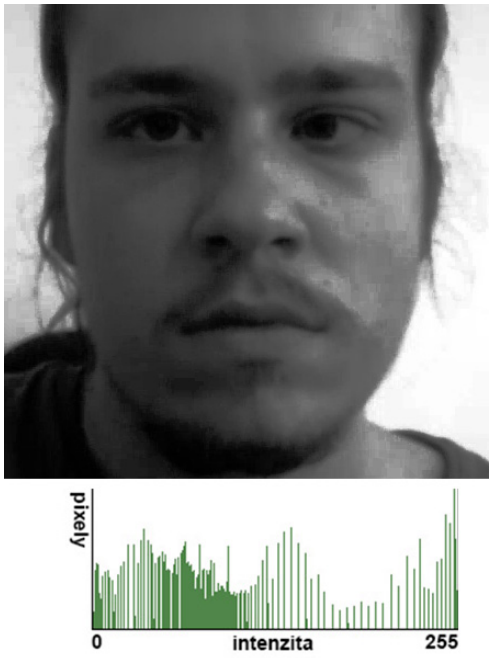
Obrázek 17: Obrázek 16 v odstínech šedi a jeho vypočtený histogram

Vypočtený histogram ukazuje, že většina pixelů má hodnotu intenzity přibližně uprostřed histogramu. Cílem tohoto algoritmu je rozprostřít dané hodnoty rovnoměrně po celé délce. To se dělá pomocí následujícího vzorce:

$$H'(i) = \sum_{0 \leq j < i} H(j) \tag{4}$$

Výsledek tohoto výpočtu se následně transformuje tak, že se H' použije pro vyhledání hodnoty, kterou má výsledný pixel (dst) obsahovat z histogramu předešlé matice (src). Neboli: $dst(x, y) = H'(src(x, y))$

Výsledný snímek i s novým histogramem je zobrazen na obrázku 18.

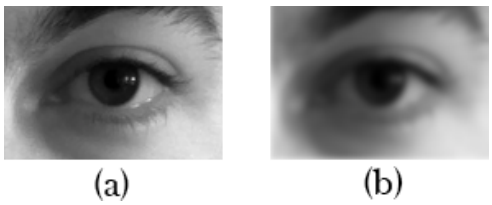


Obrázek 18: Obrázek 17 po vyrovnání histogramu

3.5 Detekce pomocí hran

První volbou při tvorbě výsledného detektoru byla detekce založená na detekování hran. Pro tento úkol byl vybrán Cannyho hranový detektor [9], který vykazuje v porovnání s ostatními detektory tohoto typu malou chybovost.

Vzhledem k tomu, že obraz získaný z kamery je málokdy perfektní, většina snímků obsahuje nepatrné množství šumu, které může znehodnotit výsledky jakékoliv detekce. Pro minimalizaci tohoto šumu byl aplikován na předpokládanou oblast oka Gaussův filtr - obr. 19.



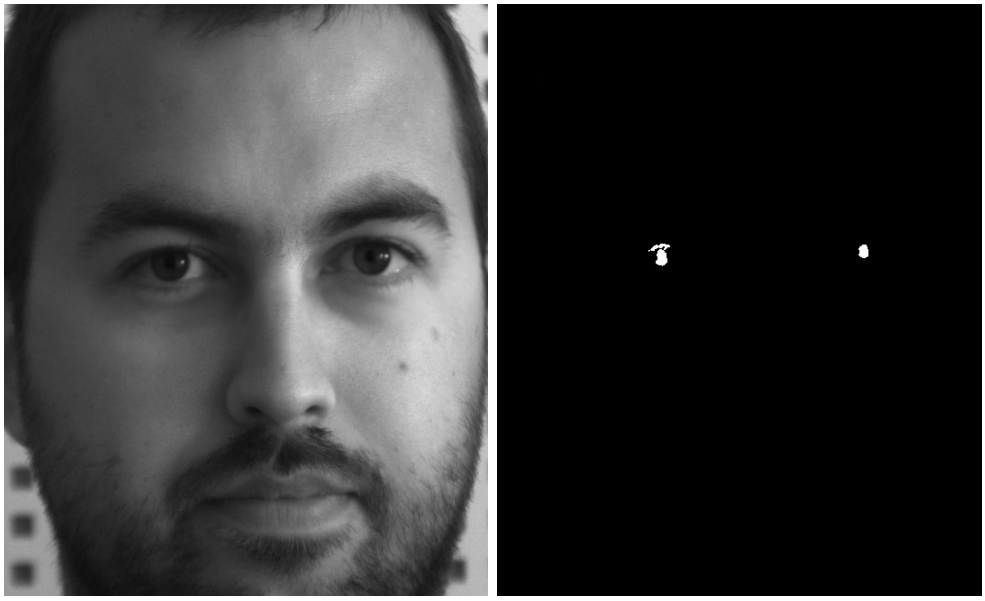
Obrázek 19: Použití Gaussova filtru. (a) - obrázek před úpravou, (b) - obrázek po použití Gaussova filtru

Tento filtr se zavedl před vyrovnáním histogramu, jehož následné zavedení způsobí, že zmizí většina přebytečných hran a zůstanou jen ty výrazné, jako je např. přestup z bělma na oční víčko.

3.5.1 Nalezení zornice

Nyní je potřeba mít jakýsi záchytný bod, okolo kterého se budou víčka vyhledávat. Tímto bodem se stává zornice díky jejímu předpokládanému tvaru a barvě. Její tvar tvoří totiž v obrazu kruhový shluk tmavých pixelů, který je při správných podmínkách možné vyhledat.

Nejprve je tedy nutné nalézt způsob, kterým lze tento kruhový tvar najít. Tento algoritmus využívá metodu, díky které se vytvoří maska určité hodnoty. Jinými slovy se vytvoří prázdná matice, do které se překopíruje jenom ta část obrazu, která zapadá mezi dvě určené hranice černobílých hodnot. Oční víčko bývá nejtemnějším bodem, proto se tedy matice originálního obrázku prohledá pro tento nejtmavší bod. Jakmile je bod nalezen, použije se jeho hodnota jako dolní hranice této masky. Pro detekci horní hranice se prozkoumá blízká oblast tohoto bodu a její hodnota je využita jako horní hranice. Nyní můžeme pomocí popsané metody vytvořit potřebnou masku. Výsledek je zobrazen na obrázku 20.



Obrázek 20: Maska daného snímku tváře.

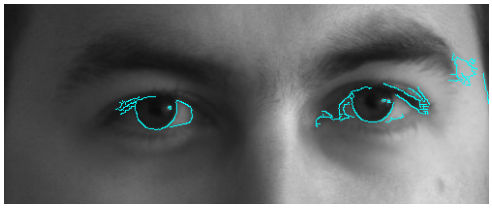
Na vytvořené masce se vytvoří dvě oblasti v pozici zornic. Tyto oblasti již vykazují jakési kruhové nebo půlkruhové vlastnosti. Algoritmus tedy využije Houghovu transformaci pro vytvoření kružnice okolo každé oblasti. Tato transformace funguje na podobném principu jako Houghova transformace pro vytvoření přímky. Zde se však nepropojují pouze body v přibližné linii, ale body se prozkoumají a hledá se jejich přibližný střed, od kterého se určí poloměr. Algoritmus tedy nyní může vypočíst a vykreslit kružnici, i když jsou některé body nepřítomné - obrázek 21.

3.5.2 Detekce hran očních víček

Po nalezení zornice algoritmus vytvoří na její pozici obdélník, pomocí kterého se později určí stav oka. Detektor to řeší tak, že se snaží horní a dolní stěnu obdélníku mít na stejných pozicích,

Obrázek 21: Tvář s vyznačenými zornicemi pomocí Houghovy transformace.

jako jsou oční víčka. Pro toto algoritmus využívá Cannyho detektor hran. Jedná se o jeden z nejvyužívanějších detektorů a byl vybrán i zde kvůli jeho nízké chybovosti a eliminaci falešných hran. Pro správné fungování potřebuje detektor určit správnou horní a spodní hranici, kde bude vyhledávat potřebné hrany. Použije se tedy podobná metoda jako při určení hranic masky, zároveň je také přidána malá hodnota pro nalezení blízkého okolí. Výsledek nám zaručí, že hrany oka se zvýrazní a lze s nimi pracovat - obrázek 22. Obdélník, který se vytvořil dříve na pozici zornice, nyní hledá jím procházející hrany. Pokud nalezne hranu, zvětší svou velikost tak, aby byla tato hrana obsažena uvnitř obdélníku. Tento proces probíhá tak dlouho, dokud nezahrne všechny hrany oka - obrázek 23.

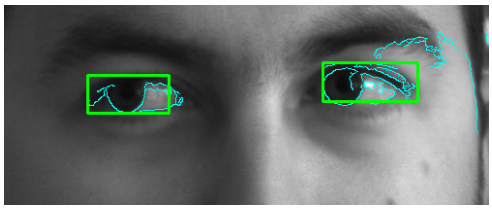


Obrázek 22: Vyznačené hrany v očích.

3.5.3 Shrnutí algoritmu

I přes dlouhé experimentování se nepodařilo dosáhnout požadovaných výsledků. Pokud video bylo v dostatečné kvalitě, algoritmus dokázal určit, v jakém stavu se oční víčka nachází. Pokud nebyla nalezena zornice, oko bylo považováno za zavřené. Pokud však zornice byla nalezena, určoval se zbytek pomocí tvaru vytvořeného obdélníku, který vznikl okolo oka.

Tento algoritmus ovšem nedosahoval dostatečné přesnosti u videí s malým rozlišením a kvůli velké závislosti na světelných podmínkách bylo těžké určit přesný tvar očních víček. Pokud například nebylo oko dostatečně nasvícené, algoritmus nezaznamenal dostatek hran, podle kterých by určil správný tvar. Výsledek lze vidět v tabulce 1.



Obrázek 23: Výsledný stav hranového detektoru. Modrý obdélník - Detekovaná výška otevření oka.

Tabulka 1: Výsledek detekce pomocí hran

	Počet snímků	Špatný tvar	Úspěšnost tvarů	Špatný stav	Úspěšnost stavů
Video 1	264	143	46%	117	56%
Video 2	396	184	54%	87	78%
Video 3	512	388	24%	361	29%

3.6 Detekce pomocí porovnání vzoru

Po selhání předešlého algoritmu byla následná metoda založena na kompletně jiném způsobu detekce. Pomocí informací nastudovaných z článku zmíněného výše, byl sestaven detektor na základě hledání a porovnávání vzorů.

Algoritmus, který se řídí pomocí vyhledávání vzorů [10], potřebuje pouze správný vzor a obrázek v originálním stavu. Proto se nezavádí základní kroky pro nic jiného než jen nalezení pozice oka.

3.6.1 Vyhledání zornice pomocí vzoru

Když je oko nalezeno, musí se opět najít určité základní body. Prvním základním bodem se stává zornice. Ta bude vyhledávána pomocí následujícího vzoru - obrázek 24.

Obrázek 24: Vzor hledané zornice.

Odpovídající oblast se najde tak, že se vzor vloží do oblasti nalezeného oka a posouvá se postupně o jeden pixel. Výsledek se ukládá do matice, která hodnotu každého prozkoumaného pixelu uloží do její odpovídající pozice. Hodnota pixelu se určí podle zvolené metody. Pro tento

způsob detekce byla zvolena metoda CV_TM_SQDIFF_NORMED. Ta vytvoří matici podle následujícího vzorce, kde **I** - zdrojový obrázek, **T** - vzor, **R** - výsledek:

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}} \tag{5}$$

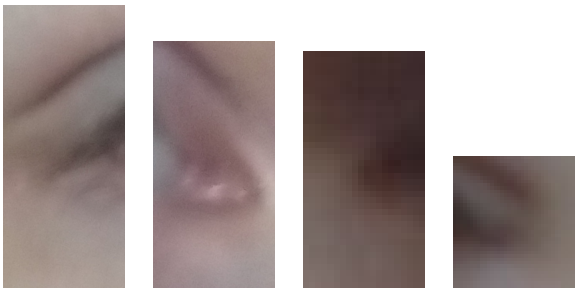
Jinými slovy čím více oblast na dané pozici odpovídá zadanému vzoru, tím menší hodnotu má. 0 se tedy rovná perfektní shodě. To znamená, že pokud chceme najít místo, kde se zornice nachází, stačí matici projít a nalézt x a y souřadnice nejtmavšího pixelu. Levá strana obrázku 25 ukazuje, jak vypadá výsledná mapa, kdyby byl vzor vyhledáván v celém obrazu. Mapa obsahuje názorně označená místa, kde se nachází shoda.

Obrázek 25: Levá strana - Zobrazení mapy porovnání vzoru. Pravá strana - Nalezení zornice pomocí TM detektoru.

Následně, aby se nalezl střed zornice, se využije výška vzoru. Souřadnice y tohoto středu bude sloužit později k vykreslení výsledného tvaru. Výsledek detekce zornic je zobrazen na pravé straně obrázku 25.

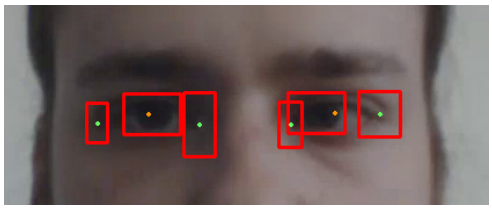
3.6.2 Nalezení koutků pomocí vzorů

Metoda pro nalezení koutků je přibližně stejná jako metoda pro nalezení zornic. Jediným problémem zde je, že algoritmus musí mít různé vzory pro různé oči. Levé oko má jiné vzory než oko pravé. Použité vzory jsou zobrazeny na obrázku 26.



Obrázek 26: Ukázka vzorů koutků pro obě oči.

Jakmile jsou nalezeny koutky očí, algoritmus vypočte jejich střed. Tyto body se nyní společně s y souřadnicí středu zornice použijí k určení středu oka. Výsledek je zobrazen na obrázku 27



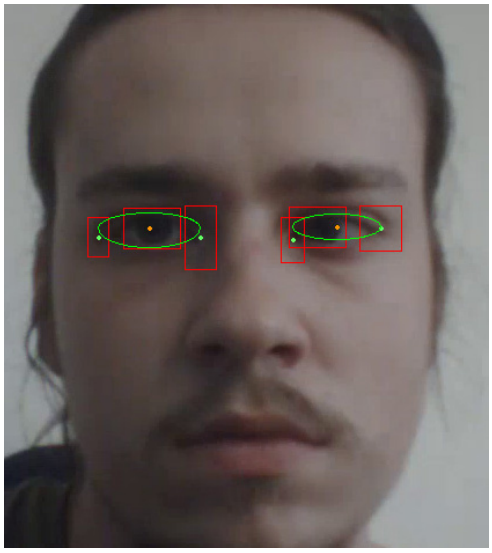
Obrázek 27: Nalezené zornice a koutky s potřebnými body pomocí vzorů.

3.6.3 Detekce tvaru očních víček pomocí červeného kanálu

Po nalezení těchto základních bodů je nyní potřeba detekovat tvar horního a spodního víčka. Toho bylo dosaženo následovně. Ze středového bodu (oranžový bod na obrázku 27) se vyvedou dva detekční body směrem nahoru a dolů, které mezi sebou porovnávají hodnotu červeného kanálu. Jakmile je rozdíl těchto dvou hodnot větší než předchozí naměřený rozdíl, pozice se uloží do nově vytvořeného bodu. Pozice tohoto nového bodu se změní na nově určenou pozici pokaždé, když se zvýší hodnota rozdílu. Jakmile je toto provedeno na obou stranách, detektor získá nejvyšší bod horního víčka a nejnižší bod víčka dolního. Tyto body slouží pro výslednou elipsu, která využije jejich vzdálenost k vytvoření svého tvaru.

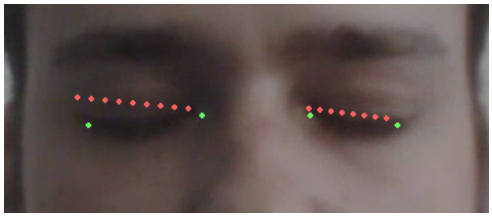
3.6.4 Určování stavů

Detektor umí rozlišovat mezi třemi stavy. Otevřený a přivřený stav se určují pomocí vzdálenosti víček od sebe v porovnání se vzdáleností koutků. To zajistí, že oči různě vzdálené od kamery budou mít stále stejnou podmínku. Stav zavřeného oka se určuje odlišně. Vzhledem k tomu, že



Obrázek 28: Zvýrazněný tvar TM detektoru.

se na lokaci zornice nedá spoléhat, když je oko zavřené (tím pádem střed elipsy nelze určit), je tento stav detekován následovně. Z pravého koutku do levého se vyvede skupina bodů, které kontrolují hodnotu pixelů na těchto místech. Pokud se hodnoty výrazně nemění (detekční body nepřechází z bělma do barev duhovky a poté zornice a naopak), pak se stav oka považuje za zavřený. Tyto body jsou zobrazeny na obrázku 29. Výsledné stavy jsou zobrazeny na obrázku 30.



Obrázek 29: Detekce zavřeného oka pomocí průzkumných bodů.

Obrázek 30: Detekce stavů detektoru porovnávání vzorů.

3.6.5 Shodnocení a efektivita detektoru

Ačkoliv byl detektor funkční, přesto nevyhovoval kvůli malé přesnosti. Kvůli faktu, že každé oko je jiné, detektor potřebuje velké množství vzorů a tím pádem i dlouhou výpočetní dobu. Všechny

vlastnosti daného detektoru ovšem nebyly nevyužitelné. Způsob určování stavů fungoval (i když ne vždy, jak je vidět na tabulce 2) a lze ho s menší úpravou využít i dále.

Detektor byl testován na videu 1 - obrázek 12, které má rozlišení 1024x768. Další videa skončila neúspěchem kvůli neodpovídajícím vzorům.

Tabulka 2: Výsledek detekce pomocí vzorů

	Počet snímků	Špatný tvar	Úspěšnost tvarů	Špatný stav	Úspěšnost stavů
Video 1	264	110	59%	90	66%

3.7 Detekce pomocí kontrastu

Tento algoritmus byl založen na zkoumání pixelů a jejich barevných hodnot. Oproti předešlým algoritmům, které se snaží nalézt tvar pomocí detekování hran, tento algoritmus není natolik závislý na světelných podmínkách a oproti detektoru založeném na porovnávání vzorů tento detektor nepotřebuje základní databázi vzorů. Níže jsou uvedeny kroky, které byly provedeny při vývoji tohoto detektoru.

3.7.1 Základní kroky

Základní kroky tohoto algoritmu jsou stejné jako u předešlých detektorů s tou výjimkou, že se zde používá mediánový filtr poté, co se obraz převede do odstínů šedi a předtím, než se vyrovná jeho histogram. Mediánový filtr nám zajistí, že případný šum nezpůsobí špatnou detekci kontrastu. Funguje tak, že hodnota každého pixelu se zprůměruje s hodnotou pixelu sousedního. Tím se zbaví míst, kde má například jeden nebo dva pixely hodnotu 0 a zanechá oblasti, kde je shluk těchto pixelů pohromadě (oblast zornice).

3.7.2 Detekce zornice

Detektor opět začíná vyhledáním zornice. Tento krok je nezbytný a funguje podobně jako u předešlé verze detektoru, která byla založena na vyhledávání hran. Obdobně i zde detektor hledá nejtmavší bod v oblasti oka. Algoritmus projde matici této oblasti a nalezne nejnižší možnou hodnotu. Jakmile je požadovaný pixel nalezen, je prozkoumána blízká oblast, zda se pixel nachází mezi ostatními body stejné nebo podobné hodnoty. Pokud ne, pak se výběr posune pod nalezený bod o pár pixelů níže a hledá se nejtmavší pixel znovu. Tato kontrola se provádí k eliminaci detekce černých pixelů v řasách nebo obočí, které mediánový filtr nedokáže odstranit.

3.7.3 Nalezení koutků a víček

Po nalezení zornice se nyní musí najít koutky očí. Z nalezeného bodu uvnitř zornice se původně vyvedly dva body do stran, které zkoumaly, zda se kontrast změnil o určitou hodnotu, která je světlejší než zornice a duhovka a zároveň tmavší než bělmo. Tato technika fungovala dostatečně

Obrázek 31: Nejtmavější body očí.

dobře v případě, že tvář nebyla příliš nasvícená. Pokud ano, pak se světlá oblast tváře rovnala bělmu v hodnotách kontrastu a bod pokračoval ve svém hledání dále, dokud neselhal.

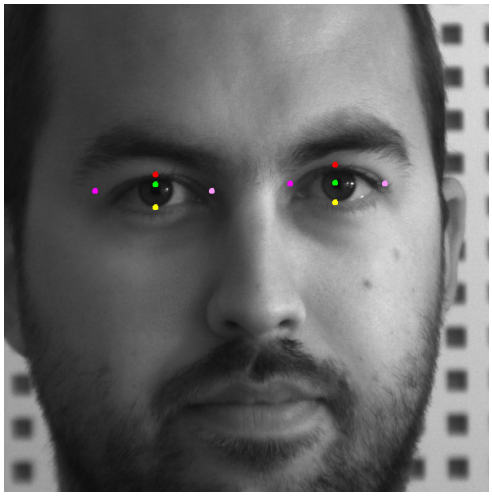
Proto byla tedy využita jiná technika. Haarův klasifikátor, který byl použit dříve na detekci oka, se snaží, aby nalezená oblast obsahovala pouze dané oko od jednoho koutku k druhému. Této vlastnosti lze využít pro nastavení x souřadnice koutku. y souřadnice bude určena později, jakmile se naleznou víčka.

Detekce víček byla opět vytvořena pomocí dvou bodů. Tento krok funguje podobně jako tomu bylo u detekce za pomocí vzorů. Z nejtmavšího bodu, který byl nalezen v zornici, se vyšlou dva body. Nejprve se vyšle spodní bod, který hledá hodnotu pixelu větší, než je zornice a který byl vypočtený ze sousední oblasti oka. Vzalo se pár náhodných pixelů a jejich hodnota se zprůměrovala a následně vydělila, aby se zmenšila výsledná hodnota kvůli chybovosti. Tím se zaručí, že se bod zastaví, jakmile detekuje odstín barvu kůže. Tato stejná hodnota se použije i pro spodní bod. Jakmile jsou oba body na správném místě, jejich střed se použije pro nalezení y souřadnice koutků.

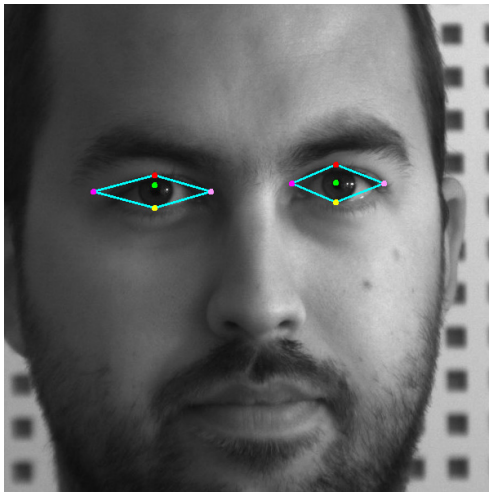
Výsledné body - obrázek 32 tvoří základ pro detekci tvaru a následných stavů snímku.

3.7.4 Zvýraznění tvaru

Dříve využité techniky využívaly vykreslení pomocí obdélníku a elipsy. Pro odlišení od těchto technik jsem se rozhodl využít tentokrát jednoduchý čtyřúhelník, jehož vrcholy odpovídají pozicím čtyř základních bodů. Jejich výsledek lze vidět na obrázku 33.



Obrázek 32: Zvýrazněné výsledné body.



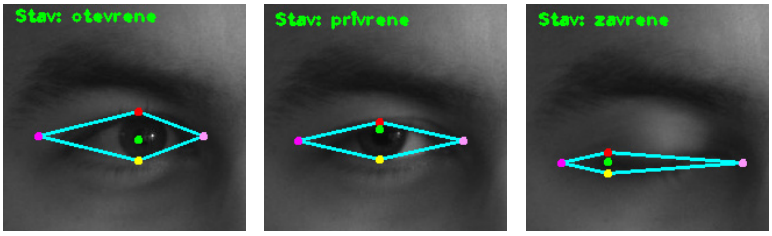
Obrázek 33: Propojené body výsledné detekce.

3.7.5 Určení stavů

Ze získaných bodů lze nyní vypočíst různé stavy oka. Tento detektor je momentálně schopný detekovat 3 stavy, otevřené, přivřené a zavřené oko. Každý stav se určuje pomocí vzdálenosti koutků, která se porovná se vzdáleností horního a dolního víčka. Tímto se zajistí, že různě velké oči budou mít stejné podmínky pro své stavy, podobně jako tomu bylo u detekce pomocí hledání vzorů. Neboli:

$$S = \frac{B_y - T_y}{R_x - L_x} \tag{6}$$

Kde **S** - stav, **B** - spodní víčko, **T** - horní víčko, **R** - pravý koutek, **L** - Levý koutek
Výsledné stavy vypadají následovně:



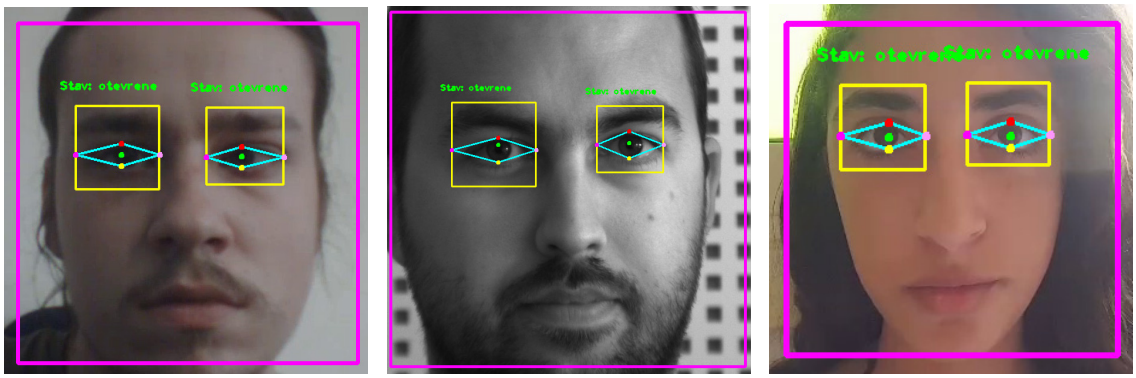
Obrázek 34: Stavy detekované změnou v kontrastu.

3.7.6 Zhodnocení výsledného detektoru

Výsledný detektor byl vytvořen po dlouhém experimentování a testování různých technik. Pokud obraz není příliš tmavý nebo zornice nejsou příliš výrazně nasvícené, dosahuje finální verze tohoto detektoru relativně velké přesnosti. V případě tmavého obrazu či přílišného nasvícení zornic by detektor nebyl schopen nalézt zornici a video by skončilo neúspěchem. Následná tabulka 3 zobrazuje dané výsledky. Výsledné snímky této detekce lze vidět na obrázku 35.

Tabulka 3: Výsledek detekce pomocí kontrastu

	Počet snímků	Špatný tvar	Úspěšnost tvarů	Špatný stav	Úspěšnost stavů
Video 1	264	51	81%	43	84%
Video 2	395	52	87%	47	88%
Video 3	512	133	74%	123	76%



Obrázek 35: Ukázky výsledků detektoru založeném na změnách kontrastu.

4 Závěr

Cílem této práce bylo vytvořit detektor, který by byl schopný zpracovat data získaná z obrazu a nalézt v nich lidské oko, ze kterého lze následně získat tvar očních víček. Nejprve se tato práce věnuje nastudování různých technik, jak je možné této detekce dosáhnout. Tyto techniky jsou detailně popsány, jejich nastudování bylo nutné k porozumění efektivního využití detekce objektů pro tento úkol.

Po nastudování dostatečného množství informací byla práce zaměřena na sestavení daného detektoru. To probíhalo pomocí experimentování na určitém množství videí, která simulovala různé světelné podmínky a kvality kamer. Než se dosáhlo přijatelných výsledků, byly vytvořeny celkem tři detektory.

První detektor byl založen na vyhledávání hran. I přes dlouhé experimentování detektor nevykazoval dostatečně kvalitní výsledky. Z těchto důvodů bylo třeba vytvořit další detektor, který k dosažení výsledku využíval již sofistikovanější algoritmy.

Druhý detektor tedy využíval techniky porovnávání vzorů. Výsledky tohoto detektoru již byly po dostatečné úpravě přijatelné. Avšak závislost tohoto detektoru na databázi vzorů, bez kterých by jinak nefungoval, způsobila, že bylo nutné najít jinou techniku.

Třetí a výsledný detektor, který byl pro tuto práci vytvořen, byl sestaven zkombinováním prvků, které úspěšně fungovaly v předešlých dvou detektorech. Funkčnost těchto algoritmů však byla ještě více doladěna, což se projevilo na zvýšené robustnosti tohoto detektoru. Princip fungování byl založen na vyhledávání změn v kontrastu rozpoznávaného snímku. Podle rozdílného kontrastu poté algoritmus nalezl tvar očních víček.

Výsledný algoritmus tedy vykazuje relativně velkou přesnost. Díky schopnosti rozlišit mezi třemi stavy oka (otevřené, přivřené, zavřené), lze tento algoritmus efektivně využívat v systémech pro detekci stavu řidiče ve vozidle. Pokud by takový systém vyhodnotil, že řidič ztrácí pozornost, mohl by na to patřičným způsobem reagovat. Implementace zařízení s takovým softwarem do vozidel by tedy mohla značně snížit počet nehod zaviněných únavou řidiče.

Literatura

[1] DING, Liya., MARTINEZ, M., Aleix. *Precise Detailed Detection of Faces and Facial Features* [online]. 2008, [cit. 2017-02-21]. Dostupné z: http://www2.ece.ohio-state.edu/~aleix/CVPR08.pdf

[2] VEZHNEVETS, Vladimir., DEGTIAREVA, Anna. *Robust and Accurate Eye Contour Extraction* [online]. 2003, [cit. 2017-02-21]. Dostupné z: http://graphicon.ru/html/2003/Proceedings/Technical/paper492.pdf

[3] WAGH, Amit., Madhukar., TODMAL, R., Satish. *Eyelids, Eyelashes Detection Algorithm and Hough Transform Method for Noise Removal in Iris Recognition.* [online]. 2015, [cit. 2017-02-21] Dostupné z: http://research.ijcaonline.org/volume112/number3/pxc3901239.pdf

[4] BERNARD, Florian., DEUTERB, Eric., Christian., GEMMARA, Peter., SCHACHINGERB, Hartmut. *Eyelid Contour Detection and Tracking for Startle Research related Eye-Blink Measurements from High-Speed Video Records.* [online]. 2013, [cit. 2017-02-21] Dostupné z: https://www.hochschule-trier.de/uploads/media/eyeblink-preprint.pdf

[5] SIROHEY, S., ROSENFELD, A., DURIC, Z. *A method of detecting and tracking irises and eyelids in video* [online]. 2001, [cit. 2017-02-21] Dostupné z: https://ai2-s2-pdfs.s3.amazonaws.com/7d48/19bb6329af6c83eb2507775f4f6107cd9649.pdf

[6] ADAM, Mathieu., ROSSANT, Florence., AMIEL, Frederic., MIKOVIKOVA, Beata., EA, Thomas. *Eyelid Localization for Iris Identification.* [online]. 2008, [cit. 2017-02-21] Dostupné z: http://www.radioeng.cz/fulltexts/2008/08_04b_082_085.pdf

[7] Pengfei Cai and Chongke Wang CAI, Pengfei., WANG, Chongke. *An Eyelid Detection Algorithm for the Iris Recognition .* [online]. 2015, [cit. 2017-02-21] Dostupné z: http://www.sersc.org/journals/IJSIA/vol9_no5_2015/11.pdf

[8] OpenCV team. *OpenCV.* [online]. 2017, [cit. 2017-02-21] Dostupné z: http://opencv.org/

[9] OpenCV team. *Canny Edge Detector - OpenCV documentation* [online]. 2017, [cit. 2017-02-21] Dostupné z: http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html

[10] OpenCV team. *Template Matching - OpenCV documentation* [online]. 2017, [cit. 2017-02-21] Dostupné z: http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html

A Příloha na CD

A.1 code - Kód výsledného algoritmu

A.2 video - Testovaná videa

A.3 results - Výsledky detekce